COMPUTE

Microprocessor Newsletter

Volume 5 Number 2 Spring 1979

A SC/MP LCDS COMPATIBLE 5V 2716 EPROM PROGRAMMER

by John R. Seal
John W. Scheer

Faced by the need for a simple, easy-to-implement way to program 5V 2716 EPROMs, we found SC/MP, with its LCDS, to be unquestionably the cleanest solution. This report describes the resulting programming card and accompanying program.

The card used was a VECTOR 4066-4 high-density wirewrapping card which is perfectly compatible with the LCDS bus structure. The card can program four 5V 2716's sequentially and can also serve as an 8K ROM card in a SC/MP system. The only non-LCDS requirement is that 25.7V ±1V be provided by an external power supply for programming (this external voltage is, of course, not necessary if the board is not being used as a programmer, but as a ROM card).

As shown in Figure 1 (see page 8), in order to program a 2716, 5V V_{CC} must be applied first, then V_{PP} (25V ±1V). The address of the byte to be programmed must be presented and OE brought high, the desired data must be presented, and, at least 2 μsec after the address has become stable, the CE/PGM input is raised to actually begin programming the address. CE/PGM must stay high for 50ms ± 5ms and then be returned low. After another interval of at least 2 μsec, OE is returned low for verification of the byte.

To examine our circuit, the high four address bits are brought to a pair of LS85s, each of which is watching half of

Continued on Page 4 . . . . .

National does not assume any responsibility for use of any circuitry described, no circuit patent licenses are implied, and National reserves the right, at any time without notice, to change said circuitry.

NIBL'ing On The SC/MP

—A few words from a hardcore user

by Erik Skovgaard
Nordlundsvej 10
2650 Hvidovre, Denmark

The Tiny BASIC interpreter for the INS8060 called NIBL (National Industrial BASIC Language) is quite a useful tool for simple control applications. It is, however, my feeling that NIBL is not very widely used, perhaps due to the lack of extensive documentation. Let me then attempt to rectify the situation by relating a few experiences with the interpreter. They lie mainly in the following areas:

Terminal I/O

The I/O routines are quite wisely placed in the end of the interpreter from 0F77 (hex) to 0FFF, thus making it possible to write your own I/O and either replace the last ROM or - in case of a MAXIROM - address decode the last section of the ROM in such a manner that 0F77 to 0FFF is ignored and your own routine is used instead.

The original NIBL is meant to run with an ASR 33 Teletype ® at 110 baud with ≥ 2 stop bits which - since the delays are software generated - causes programs with TTY outputs to execute rather slowly. Tables 1 and 2 show the necessary changes in the interpreter to let it function at bit rates of 300, 600 and 1200 baud. Note the difference between SC/MP I and SC/MP II in this respect. The former is assumed to run with a 1MHz crystal and the latter with a 4MHz crystal.

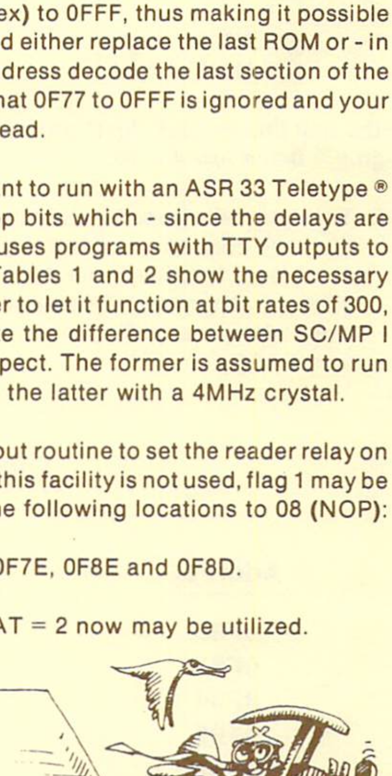
Flag 1 is used by the output routine to set the reader relay on the paper tape reader. If this facility is not used, flag 1 may be liberated by changing the following locations to 08 (NOP):

0F7B, 0F7C, 0F7D, 0F7E, 0F8E and 0F8D.

Thus, the command STAT = 2 now may be utilized.

REQUALIFICATION ISSUE

To continue receiving COMPUTE, the Requalification Form in this issue must be completed and returned. If you don't, you won't be getting COMPUTE anymore, and we wouldn't want that to happen.



Sponsored by National Semiconductor Corporation, 2900 Semiconductor Drive, Santa Clara, California 95051

Program Storage Format

The program lines in NIBL are stored as follows:

1. byte : Line # high
2. byte :Line # low
3. byte :Total line length (including # and CR)
4. thru Nth byte:The statement line in ASCII format.
   (N + 1) th byte :CR (= 0D hex)

The first two bytes in each 4K page contain FF and 0D followed by the actual lines with their parameters. Additionally, the last line in a page is followed by two bytes of FF (= an invalid line # since NIBL only allows line numbers in the range 0 thru 32767) immediately after the last CR.

The first line (including the leading FF, 0D) in page 1 starts at 111E, while all the rest of the valid pages (2 through 7) start at N000. Whenever NIBL is initialized, after a reset for example, it automatically stores FF in the third and fourth position of each page, thus overwriting the first and the second byte in the first line, producing an invalid line number or an end-ofprogram flag. The same thing happens with the command "NEW". So we may conclude the following:

1.) After RESET or an accidental "NEW", a program in RAM may be retrieved by typing:

For page 1:

```
@ # 1120 = high #
@ # 1121 = low #
```

For page N:

```
@ # N002 = high #
@ # N003 = low #
```

If the first line in your "lost" program was number 10 then high # = 0 and low # = 10.

2.) The automatic restart feature that tests for the presence of a valid program in page 2 requires that page 2 is ROM or at least that hardware prevents WRITE during the startup sequence. Otherwise the first line # is overwritten and NIBL proceeds to prompt with a ">", with PAGE = 2.

Linking Subroutines

The addition of machine-language subroutines provides NIBL with the proper flexibility to be used together with highspeed peripherals and also makes it possible to extend the reach of the - otherwise rather limited - instruction set. The statement

```
LINK     #2E00
```

loads P3 with 2E00 and does a LD @ -1 (3) adjusting P3 to start execution at the proper address with an XPPC P3. Thus no "NOP" is necessary at the beginning of the subroutine. P2 is set to NIBL scratch pad (= 101C) at this point. Although most of its uses are documented in the NIBL source listing, a few additional numbers are nice to have in mind when you use this pointer. It may prove useful to access the variables from the subroutine. See table 3 for the associated addresses. It is also possible to use P2 relative RAM addresses in the subroutine without interfacing with NIBL scratch pad or variables. The range -29 through -127 (decimal) is not used by the interpreter, but note that wraparound occurs so that P2 -29 equals 1FFF, and P2 -127 equals 1F9D. The NIBL program should therefore not go all the way to the bottom of page 1, if you want to make use of the whole range of P2. All registers may be modified in the machine-language subroutine, but if P3 is changed the absolute return address 0970 (hex) must be used for reentry into NIBL instead of the otherwise obvious XPPC P3.

Real-life Programming

Finally a word about parentheses. When using the conditional "IF" statement some caution should be taken. For instance, consider the three ways of writing the following:

```
(1) IF G>1000 or G<100 GOTO 120
(2) IF (G>1000 or G<100) GOTO 120
(3) IF ((G>1000) or (G<100)) GOTO 120
```

At first glance they don't seem functionally different, but when executed the first statement only tests for the first condition (G>1000), and (2) produces a syntax error. Only the last statement works properly.

Table 1. NIBL SC/MP I Baud Rate Changes

Address	110 Baud	300 Baud	600 Baud	1200 Baud
0F85	57	A0	F0	60
0F87	04	01	00	00
0F94	85	E6	48	7D
0F96	08	02	01	00
0FB9	08	03	02	01
0FC4	FF	FF	00	C8
0FC6	17	03	08	00
0FD0	8A	F0	50	80
0FD2	08	02	01	00

Table 2. NIBL SC/MP II Baud Rate Changes

Address	110 Baud	300 Baud	600 Baud	1200 Baud
0F85	C3	29	8A	BB
0F87	08	03	01	00
0F94	45	11	D4	34
0F96	11	06	02	01
0FB9	11	06	03	01
0FC4	BB	6C	2D	99
0FC6	2F	06	03	01
0FD0	54	21	E5	44
0FD2	11	06	02	01

Table 3. Variables and Associated Addresses

	Address*	
Variable	Absolute	Relative to P2 (dec.)
A	104F	+51
B	104D	+49
C	104B	+47
D	1049	+45
E	1047	+43
F	1045	+41
G	1043	+39
H	1041	+37
I	103F	+35
J	103D	+33
K	103B	+31
L	1039	+29
M	1037	+27
N	1035	+25
O	1033	+23
P	1031	+21
Q	102F	+19
R	102D	+17
S	102B	+15
T	1029	+13
U	1027	+11
V	1025	+9
W	1023	+7
X	1021	+5
Y	101F	+3
Z	101D	+1

*The addresses shown are for the high byte of the variable. The low byte is located one address below.

an 8PST DIP switch for its page assignment. Thanks to SC/MP's convenient timing, the page selection test is completed before a read or write strobe is initiated, so we can latch which (if either) page is selected with the low 12 address bits to simplify decoding which (if any) EPROM outputs should be enabled. Address bit 12 is redundantly latched just because it might come in handy at some future time. The MODE SELECTION truth table (Figure 2, see page 8) shows the actual requirements for all 2716 operations. We keep all OEs high unless actually reading from the ROM, which causes all 2716's that are not being accessed to be in the power-saving mode. To read a 2716, OE and CE/PGM are driven low simultaneously (FLAG 1 should never be high in READ mode). We return to READ mode for verification, which offers an additional function test of the ROM.

In programming, we write the data to the desired address (either a read or write will latch the address). Data is latched at the trailing edge of NWRS when NCSEL (low true Card SELect) is low. FLAG 1 is then raised and it:

(1) TRI-STATEs the output buffers from the READ mode half of the LS139,

(2) Enables the output buffers for the PROGRAM mode half of the LS139,

(3) Enables the data latch's outputs,

(4) Disables the data buffer to prevent jamming the data bus,

(5) Turns on the 2N2905 to bring the V_{PP} line to 25.0 volts (note the 0.6V drop across the 1N4005 and 0.1V drop across the 2N2905).

After allowing a generous DELAY (of more than the minimum 2μsec) for all of the FLAG 1 functions to settle, FLAG 2 is raised, the appropriate CE/PGM line is driven high and DELAY is again invoked to provide the 50ms programming period. FLAG 2 is reset low, and then FLAG 1 is reset (the 2μsec delay is more than covered) and the byte is read for verification.

The absense of FLAG 1 TRI-STATEs the data latch and the READ strobe latches the address and page selection and completes the enabling of the 81LS95 data output buffer. MSL and MEMSEL are driven low by card selection to accommodate the requirements of the SC/MP Application Card. It should be noted that the 2N2905 is a generalpurpose PNP transistor in a metal T0-5 case; a 2N3906 was tried initially, it could not handle the programming current, broke down, and destroyed a 2716 by cremating it.

It was found that SC/MP was ideally suited to this project because of the "DLY" instruction. This allowed easy generation of all timing necessary for programming the 2716.

```
                          .TITLE P2716, 'ROUTINE TO PROGRAM 2716(EPROM)
               ;*****************************************************************
               ;*
               ;*   THIS PROGRAM IS USED TO PROGRAM A 2716 EPROM.
               ;*
               ;*   IT IS USED WITH NATIONAL'S LCDS SYSTEM AND A CARD DESIGNED
               ;*   BY THE AUTHORS.  IT USES SEVERAL PROGRAMS FROM THE LCDS
               ;*   PACKAGE AS UTILITIES.  ALL JUMPS WITHIN THE PROGRAM ARE
               ;*   RELATIVE SO IT CAN BE RUN FROM ANYWHERE IN MEMORY.
               ;*
               ;*   WHEN STARTED, THE PROGRAM WILL PROMPT WITH '/'.  INPUT IS
               ;*   IN THE FORM:
               ;*          /1000:17FF:2000
               ;*   WHERE 1000 = THE STARTING ADDRESS IN HEX, 17FF = THE END
               ;*   ADDRESS, AND 2000 = THE STARTING ADDRESS OF THE ROM.
               ;*
               ;*   THE PROGRAM VERIFIES EACH LOCATION AFTER IT IS PROGRAMED
               ;*   AND WILL REPORT ANY ERRORS ENCOUNTERED.
               ;*
               ;*****************************************************************
               ;
               ;
               ;    ASSIGNMENTS FOR EXTERNAL PROGRAMS
               ;
        7AE2   PUTC    =   X'7AE2
        7A91   GECHO   =   X'7A91
        7B50   GHEX    =   X'7B50
        7BB4   PHEX    =   X'7BB4
        7B17   MESG    =   X'7B17
               ;
               ;    PROGRAM CONSTANTS
               ;
        0001   P1      =   1
        0002   P2      =   2
        0003   P3      =   3
        FF80   EREG    =   -128                             ;SET FOR DISP ON EXT REG
               ;
   0000                   .=0
```

```
0000 08                 NOP
0001 C477   START:      LDI     077          ;SET P2 FOR SCRATCH PAD
0003 36                 XPAH    P2
0004 C420               LDI     020
0006 32                 XPAL    P2
0007 C47A               LDI     H(PUTC)      ;SET P3 TO PRINT PROMPT
0009 37                 XPAH    P3
000A C4E1               LDI     L(PUTC)-1
000C 33                 XPAL    P3
000D C40D               LDI     0D           ;PRINT CR, LF & /
000F 3F                 XPPC    P3
0010 C40A               LDI     0A
0012 3F                 XPPC    P3
0013 C42F               LDI     '/'
0015 3F                 XPPC    P3
0016 C47B               LDI     H(GHEX)      ;SET P3 TO GET HEX INPUT
0018 37                 XPAH    P3
0019 C44F               LDI     L(GHEX)-1
001B 33                 XPAL    P3
001C 3F                 XPPC    P3           ;GET STARTING ADDRESS
001D C601               LD      @1(P2)       ;  PUT IN P1
001F 35                 XPAH    P1
0020 C601               LD      @1(P2)
0022 31                 XPAL    P1
0023 40                 LDE
0024 E420               XRI     020          ;TERMINATOR = SPACE?
0026 9CD9               JNZ     START        ;NO, GO TRY AGAIN
0028 3F                 XPPC    P3           ;GET HIGH ADDRESS
0029 40                 LDE
002A E420               XRI     020          ;TERMINATOR = SPACE?
002C 9CD3               JNZ     START        ;NO, GO TRY AGAIN
002E 3F                 XPPC    P3           ;GET ROM ADDRESS
002F C601               LD      @1(P2)       ;  AND PUT IN P3
0031 37                 XPAH    P3
0032 C601               LD      @1(P2)
0034 33                 XPAL    P3
0035 C100   LOOP:       LD      (P1)         ;GET A BYTE FROM RAM
0037 CB00               ST      (P3)         ;PUT IN ROM BUFFER
0039 06                 CSA                  ;SET FLAG 1
003A DC02               ORI     2
003C 07                 CAS
003D 8F00               DLY     0            ;ALLOW FLAG TO SETTLE
003F 06                 CSA                  ;SET FLAG 2
0040 DC04               ORI     4
0042 07                 CAS
0043 C440               LDI     64           ;HOLD PGM SIGNAL FOR 50 MS
0045 8F61               DLY     97
0047 06                 CSA                  ;CLEAR FLAG 2
0048 E404               XRI     4
004A 07                 CAS
004B E402               XRI     2            ;THEN CLEAR FLAG 1
004D 07                 CAS
004E C300               LD      (P3)         ;READ ROM
0050 E100               XOR     (P1)         ;COMPARE TO RAM
0052 982B               JZ      OK           ;IF THEY AGREE, CONTINUE
0054 C47B               LDI     H(MESG)      ;ELSE PRINT ERROR MESSAGE
0056 37                 XPAH    P3
0057 CA03               ST      3(P2)        ;AND SAVE CURRENT ADDRESS
0059 C416               LDI     L(MESG)-1
005B 33                 XPAL    P3
005C CA04               ST      4(P2)
005E 3F                 XPPC    P3           ;GO PRINT MESSAGE
005F 0098               .DBYTE  ERMSG
0061 C47B               LDI     H(PHEX)      ;PRINT ADDRESS IN HEX
0063 37                 XPAH    P3
0064 C4B3               LDI     L(PHEX)-1
0066 33                 XPAL    P3
```

```
0067 C203          LD      3(P2)          ;GET HIGH BYTE OF ADDRESS
0069 3F            XPPC    P3             ;PRINT IT
006A C204          LD      4(P2)          ;GET LOW BYTE
006C 3F            XPPC    P3             ;PRINT IT

006D C47A          LDI     H(PUTC)        ;P3 = PUTC
006F 37            XPAH    P3
0070 C4E1          LDI     L(PUTC)-1
0072 33            XPAL    P3
0073 C40D          LDI     0D             ;PRINT CR,LF
0075 3F            XPPC    P3
0076 C40A          LDI     0A
0078 3F            XPPC    P3
0079 C203          LD      3(P2)          ;RESET P3 TO ROM
007B 37            XPAH    P3
007C C204          LD      4(P2)
007E 33            XPAL    P3

007F 31     OK:    XPAL    P1             ;SEE IF DONE
0080 01            XAE
0081 40            LDE
0082 31            XPAL    P1
0083 40            LDE
0084 E201          XOR     1(P2)
0086 9C0A          JNZ     BUMP           ;IF NOT DONE, GO BUMP PTR'S
0088 35            XPAH    P1
0089 01            XAE
008A 40            LDE
008B 35            XPAH    P1
008C 40            LDE
008D E200          XOR     (P2)
008F 9C01          JNZ     BUMP
0091 00            HALT                   ;DONE, SO STOP
0092 C701   BUMP:  LD      @1(P3)         ;BUMP ROM POINTER
0094 C501          LD      @1(P1)         ;GET NEXT BYTE AND BUMP PTR
0096 909D          JMP     LOOP           ;GO DO NEXT BYTE

0098 4552   ERMSG: .ASCII  'ERROR @ '
009A 524F
009C 5220
009E 4020
00A0 00            .BYTE   0
     0000          .END

 ******      0 ERRORS IN ASSEMBLY    ******


SC/MP ASSEMBLER REV-C 02/06/76, MODCOMP VERSION 11/20/78          3/24/1979  12:
P2716


BUMP    EREG    ERMSG   GECHO   GHEX    LOOP    MESG    OK      P1      P2
0092    FF80    0098    7A91    7B50    0035    7B17    007F    0001    0002

P3      PHEX    PUTC    START
0003    7BB4    7AE2    0001


   SOURCE CHECKSUM   6A7A   OBJECT CHECKSUM   03D3
```

COMPUTE requalification form

This form must be returned by June 1, 1979 for you to continue receiving *COMPUTE*. Please send the form to the appropriate address (U.S., Germany, Australia, Japan, or Hong Kong) listed on the reverse side.

Attach address label here—

Note Corrections here—

____

____

____

Please answer the questions below.

I. Do you wish to receive the next volume of *COMPUTE*?____________

II. How long have you been receiving *COMPUTE*?____________

III. My title/function:

a. □ General/Corporate/Technical management

b. □ Design and development engineering/programming

c. □ Engineering services (evaluation, quality control, reliability, standards, tests)

d. □ Purchasing and procurement

e. □ Marketing and sales

f. □ Consultant or educator

g. □ Student or hobbyist

h. □ Other

IV. Check all that apply.

A. Do you buy i. □ components j. □ boards k. □ systems

B. Do you l. □ use them m. □ resell them as part of your product

V. In the past 6 months have you purchased microprocessor related components? n. □ yes o. □ no

If yes, please answer:

What type of microprocessor or related items have you purchased?	Volume over last 6 months
A.______________________________	____________
B.______________________________	____________
C.______________________________	____________
D.______________________________	____________
E.______________________________	____________
F.______________________________	____________

VI. Do you or your company own any of the following?

p. □ UDS (PACE DOS) q. □ PACE-P w/o DOS r. □ PACE LCDS

s. □ IMP-16P DOS t. □ IMP-16P w/o DOS u. □ SC/MP LCDS

v. □ SC/MP KIT w. □ BLC-80/10 x. □ SBC-80/10

VII. How often do you read *COMPUTE*?

y. □ Always z. □ Usually 1. □ Occasionally

What features/articles/notices would you like to see more of in future issues of *COMPUTE*?

____

____

____

VIII. Comments______________________________

____

____

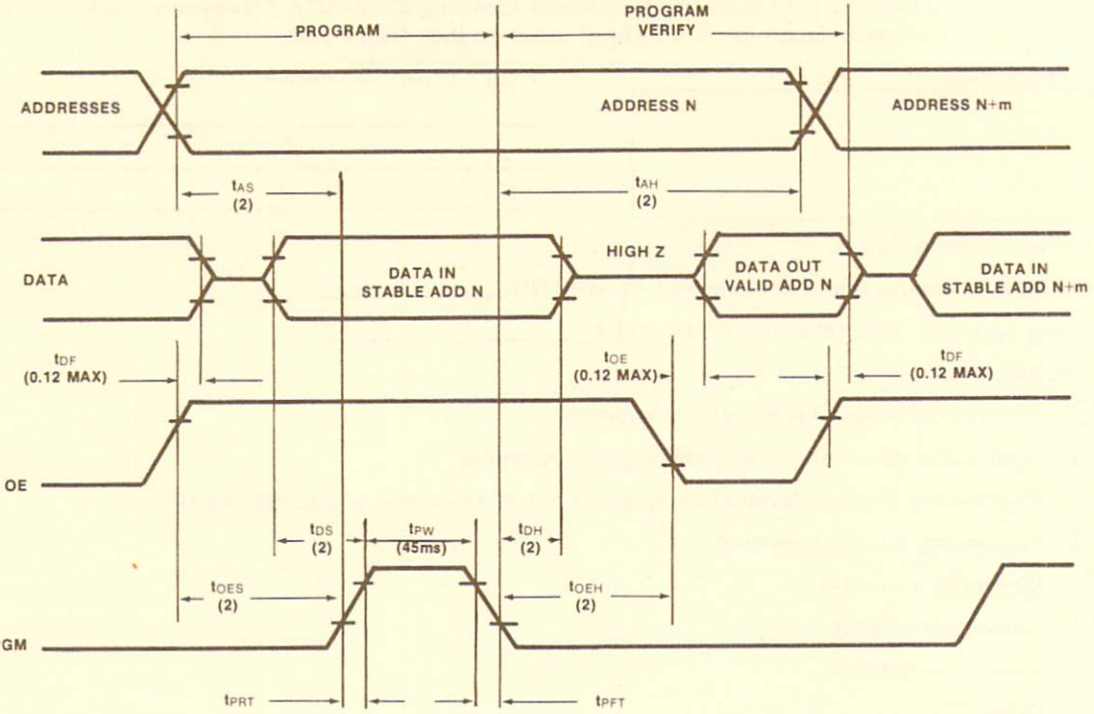


FIGURE 1. Programming Waveforms

PINS MODE	CE/PGM (18)	OE (20)	V_{PP} (21)	V_{CC} (24)	OUTPUTS (9-11, 13-17)
Read	V_{IL}	V_{IL}	+5	+5	D_{OUT}
Standby	V_{IH}	Don't Care	+5	+5	High Z
Program	Pulsed V_{IL} to V_{IH}	V_{IH}	+25	+5	D_{IN}
Program Verify	V_{IL}	V_{IL}	+25	+5	D_{OUT}
Program Inhibit	V_{IL}	V_{IH}	+25	+5	High Z

FIGURE 2. Mode Selection

Schematics for the SC/MP Compatible 5V, 2716 EPROM Programmer can be obtained by writing Scheer Electronics, 2018-C 24th Street, Los Almos, New Mexico 87544. A nominal charge of $2.00 covering postage and handling is to be enclosed with each order.

UNITED STATES
COMPUTE/5226
National Semiconductor Corp.
2900 Semiconductor Drive
Santa Clara, CA. 95051

EUROPE
COMPUTE
National Semiconductor GmbH
808 Fuerstenfeldbruck
Industriestrasse 10, West Germany

AUSTRALIA
COMPUTE
NS Electronics Pty. Ltd.
Cnr. Stud Rd. & Mtn. Highway
Bayswater, Victoria 3153
Australia

HONG KONG
COMPUTE
NS Electronics (HK) Ltd.
4 Hing Yip St., 11th Floor
Kwun Tong, Kowloon, Hong Kong

JAPAN
COMPUTE
NS International Inc.
Miyake Bldg. 6F, 1-9 Yotsuya
Shinjuku-ku, Tokyo 160, Japan